

EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT

```
WW      WW  RRRRRRRR  IIIIII  EEEEEEEEE  DDDDDDDD  TTTTTTTTT  MM      MM  SSSSSSSS  GGGGGGGG
WW      WW  RRRRRRRR  IIIIII  EEEEEEEEE  DDDDDDDD  TTTTTTTTT  MM      MM  SSSSSSSS  GGGGGGGG
WW      WW  RR      RR  II      EE      DD      DD  TT      MMMM  MMMM  SS      GG      GG
WW      WW  RR      RR  II      EE      DD      DD  TT      MMMM  MMMM  SS      GG      GG
WW      WW  RR      RR  II      EE      DD      DD  TT      MM  MM  MM  SS      GG      GG
WW      WW  RRRRRRRR  II      EEEEEEEE  DD      DD  TT      MM      MM  SSSSSS  GG      GG
WW      WW  RRRRRRRR  II      EEEEEEEE  DD      DD  TT      MM      MM  SSSSSS  GG      GG
WW      WW  RR  RR  II      EE      DD      DD  TT      MM      MM  SS      GG      GGGGGG
WW      WW  RR  RR  II      EE      DD      DD  TT      MM      MM  SS      GG      GGGGGG
WWW     WWW  RR      RR  II      EE      DD      DD  TT      MM      MM  SS      GG      GG
WWW     WWW  RR      RR  II      EE      DD      DD  TT      MM      MM  SS      GG      GG
WW      WW  RR      RR  IIIIII  EEEEEEEEE  DDDDDDDD  TT      MM      MM  SSSSSSSS  GGGGGG
WW      WW  RR      RR  IIIIII  EEEEEEEEE  DDDDDDDD  TT      MM      MM  SSSSSSSS  GGGGGG

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```

```
0001 0 %TITLE 'EDT$WRIEDTMSG - write VMSMSG.MSG'
0002 0 MODULE EDT$WRIEDTMSG (
0003 0 IDENT = 'V04-000',
0004 0 MAIN = EDT$WRIEDTMSG
0005 0 ) =
0006 1 BEGIN
0007 1
0008 1 *****
0009 1 *
0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 * ALL RIGHTS RESERVED.
0013 1 *
0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 * TRANSFERRED.
0020 1 *
0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 * CORPORATION.
0024 1 *
0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1
0032 1 ++
0033 1 FACILITY: EDT -- The DEC Standard Editor
0034 1
0035 1 ABSTRACT:
0036 1
0037 1 This module, WRIEDTMSG.FOR, is a FORTRAN program that writes
0038 1 the file VMSMSG.MSG, which is read by the message compiler
0039 1 to produce EDT's run-time messages.
0040 1
0041 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0042 1
0043 1 AUTHOR: John Sauter, CREATION DATE: 23-Jul-1981
0044 1
0045 1 MODIFIED BY:
0046 1
0047 1 1-001 - Original, from BASMSG.FOR, created November 3, 1978, last
0048 1 revised September 24, 1979 (version 1-015). JBS 28-Jul-1981
0049 1 1-002 - Don't omit the first message. JBS 03-Aug-1981
0050 1 1-003 - Change output file name to VMSMSG.MSG. JBS 03-Aug-1981
0051 1 1-004 - Fix output file's module name. JBS 06-Aug-1981
0052 1 1-005 - Recoded in BLISS since VMS doesn't like its components to be
0053 1 dependent upon Fortran. JBS 22-Oct-1981
0054 1 1-006 - Change output file name to VMSMSG.TMP. BLS 6-May-1983
0055 1 1-007 - Correct the module header and trailer. JBS 09-May-1983
0056 1
0057 1
```



```
59 0058 1 %SBTTL 'Declarations'
60 0059 1
61 0060 1 SWITCHES:
62 0061 1
63 0062 1
64 0063 1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
65 0064 1
66 0065 1
67 0066 1 LINKAGES:
68 0067 1
69 0068 1 NONE
70 0069 1
71 0070 1 TABLE OF CONTENTS:
72 0071 1
73 0072 1
74 0073 1 FORWARD ROUTINE
75 0074 1 EDT$WRIEDTMSG, ! Write VMSMSG.TMP
76 0075 1 WRITE_FILE, ! Actually write the text
77 0076 1 PRINT, ! Print a line of text
78 0077 1 HEX_TEXT, ! Convert binary to hexadecimal
79 0078 1 PRINTABLE_TEXT; ! Convert binary to ASCII, printable
80 0079 1
81 0080 1
82 0081 1 INCLUDE FILES:
83 0082 1
84 0083 1
85 0084 1 REQUIRE 'EDT$SRC:PSECTS.REQ'; ! Define PSECT declaration macros
86 0189 1
87 0190 1 REQUIRE 'EDT$SRC:SYSSYM.REQ'; ! Define system symbols
88 0220 1
89 0221 1
90 0222 1 MACROS:
91 0223 1
92 0224 1 NONE
93 0225 1
94 0226 1 EQUATED SYMBOLS:
95 0227 1
96 0228 1
97 0229 1 LITERAL
98 0230 1 EDT$K_FAC_NO = 133; ! Facility number, for signaling.
99 0231 1
100 0232 1
101 0233 1 FIELDS:
102 0234 1
103 0235 1 NONE
104 0236 1
105 0237 1 STRUCTURES:
106 0238 1
107 0239 1 NONE
108 0240 1
109 0241 1 PSECTS:
110 0242 1
111 0243 1 DECLARE_PSECTS (EDT); ! Declare PSECTs for EDT$ facility
112 0244 1
113 0245 1 OWN STORAGE:
114 0246 1
115 0247 1 NONE
```

EDT\$WRIEDTMSG
V04-000

EDT\$WRIEDTMSG - write VMSMSG.MSG
Declarations

C 3
16-Sep-1984 02:18:31
14-Sep-1984 12:25:55

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]WRIEDTMSG.B32;1

Page 3
(2)

```

: 116      0248 1 |
: 117      0249 1 | EXTERNAL REFERENCES:
: 118      0250 1 |
: 119      0251 1 |
: 120      0252 1 | EXTERNAL ROUTINE
: 121      0253 1 |   STR$COPY DX,
: 122      0254 1 |   STR$CONCAT,
: 123      0255 1 |   LIB$GET INPUT,
: 124      0256 1 |   STR$COPY R,
: 125      0257 1 |   STR$FREE DX,
: 126      0258 1 |   EDT$MSGTXT;
: 127      0259 1 |

```

! Copy a string, by descriptor
! Concatenate strings
! Get a line from SYS\$INPUT
! Copy a string, by reference
! Free a dynamic string
! Return the text of a message

EDT\$
V04-

```
129 0260 1 %SBTTL 'Package of macros for string processing'
130 0261 1 !+
131 0262 1 ! Macro to initialize a dynamic descriptor.
132 0263 1 !-
133 0264 1
134 0265 1 MACRO
135 M 0266 1 INIT_DESCRIPTOR (DESCR) =
136 M 0267 1 DESCR [DSC$W_LENGTH] = 0;
137 M 0268 1 DESCR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
138 M 0269 1 DESCR [DSC$B_CLASS] = DSC$K_CLASS_D;
139 M 0270 1 DESCR [DSC$A_POINTER] = 0;
140 0271 1 !
141 0272 1 ! <BLF/MACRO>
142 0273 1 !+
143 0274 1 ! Macro to discard a dynamic descriptor.
144 0275 1 !-
145 M 0276 1 DISCARD_DESCRIPTOR (DESCR) =
146 M 0277 1 BEGIN
147 M 0278 1
148 M 0279 1 LOCAL
149 M 0280 1 FREE_STATUS;
150 M 0281 1
151 M 0282 1 FREE_STATUS = STR$FREE1_DX (DESCR);
152 M 0283 1
153 M 0284 1 IF ( NOT .FREE_STATUS) THEN SIGNAL_STOP (.FREE_STATUS);
154 M 0285 1
155 M 0286 1 END;
156 0287 1 !
157 0288 1 !+
158 0289 1 ! Macro to build a text line using FAO. This is a convenience macro.
159 0290 1 !-
160 M 0291 1 BUILD_TEXT_LINE (DESCR, CTL_STRING, FAO_ARGS) =
161 M 0292 1 BEGIN
162 M 0293 1
163 M 0294 1 LOCAL
164 M 0295 1 FAO_STATUS,
165 M 0296 1 COPY_STATUS;
166 M 0297 1
167 M 0298 1 CTL_STR_DSC [DSC$W_LENGTH] = %CHARCOUNT (CTL_STRING);
168 M 0299 1 CTL_STR_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
169 M 0300 1 CTL_STR_DSC [DSC$B_CLASS] = DSC$K_CLASS_S;
170 M 0301 1 CTL_STR_DSC [DSC$A_POINTER] = CH$PTR (UPLIT (CTL_STRING));
171 M 0302 1 FAO_STATUS = $FAO ?
172 M 0303 1 CTL_STR_DSC,
173 M 0304 1 OUT_LENGTH,
174 M 0305 1 TEMP_STR_DSC,
175 M 0306 1 %REMOVE (FAO_ARGS));
176 M 0307 1
177 M 0308 1 IF ( NOT .FAO_STATUS) THEN SIGNAL_STOP (.FAO_STATUS);
178 M 0309 1
179 M 0310 1 COPY_STATUS = STR$COPY_R (DESCR, OUT_LENGTH, .TEMP_STR_DSC [DSC$A_POINTER]);
180 M 0311 1 .COPY_STATUS
181 M 0312 1 END
182 0313 1 !
183 0314 1 !+
184 0315 1 ! Macro to format and print a line. Errors are returned to the caller.
185 0316 1 ! This is a convenience macro.
```



```
.. 186      0317 1  !-  
.. 187      M 0318 1  PRINT_LINE (TEXT, VARS) =  
.. 188      M 0319 1  BEGIN  
.. 189      M 0320 1  
.. 190      M 0321 1  LOCAL  
.. 191      M 0322 1  BUILD_STATUS,  
.. 192      M 0323 1  PRINT_STATUS;  
.. 193      M 0324 1  
.. 194      M 0325 1  BUILD_STATUS = BUILD_TEXT_LINE (LINE_DESC, %STRING (%REMOVE (TEXT)), VARS);  
.. 195      M 0326 1  
.. 196      M 0327 1  IF ( NOT .BUILD_STATUS) THEN RETURN (.BUILD_STATUS);  
.. 197      M 0328 1  
.. 198      M 0329 1  PRINT_STATUS = PRINT (.OUTPUT_RAB, LINE_DESC);  
.. 199      M 0330 1  
.. 200      M 0331 1  IF ( NOT .PRINT_STATUS) THEN RETURN (.PRINT_STATUS);  
.. 201      M 0332 1  
.. 202      M 0333 1  END  
.. 203      0334 1  X;  
.. 204      0335 1
```

```
206 0336 1 %SBTTL 'EDT$WRIEDTMSG - Write VMSMSG.TMP'
207 0337 1 ROUTINE EDT$WRIEDTMSG ! Write VMSMSG.TMP
208 0338 1 =
209 0339 1
210 0340 1 ++
211 0341 1 FUNCTIONAL DESCRIPTION:
212 0342 1
213 0343 1 This routine writes the file VMSMSG.TMP.
214 0344 1
215 0345 1 CALLING SEQUENCE:
216 0346 1
217 0347 1 ret_status.wlc.v = EDT$WRIEDTMSG ()
218 0348 1
219 0349 1 FORMAL PARAMETERS:
220 0350 1
221 0351 1 NONE
222 0352 1
223 0353 1 IMPLICIT INPUTS:
224 0354 1
225 0355 1 NONE
226 0356 1
227 0357 1 IMPLICIT OUTPUTS:
228 0358 1
229 0359 1 NONE
230 0360 1
231 0361 1 COMPLETION STATUS:
232 0362 1
233 0363 1 $$$_NORMAL Normal successful completion
234 0364 1 Any error from LIB$GET_INPUT or STR$FREE1_DX
235 0365 1
236 0366 1 SIDE EFFECTS:
237 0367 1
238 0368 1 Writes a file.
239 0369 1 Any errors from RMSS$CREATE, RMSS$OPEN, RMSS$CONNECT or RMSS$CLOSE
240 0370 1 are signalled.
241 0371 1
242 0372 1 --
243 0373 1
244 0374 2 BEGIN
245 0375 2
246 0376 2 LOCAL
247 0377 2 OUTPUT_BUFFER : BLOCK [132, BYTE], ! output buffer, for RMS
248 0378 2 OUTPUT_FAB : $FAB_DECL, ! RMS FAB for the output file
249 0379 2 OUTPUT_NAM : $NAM_DECL, ! RMS NAM for the output file
250 0380 2 OUTPUT_RAB : $RAB_DECL, ! RMS RAB for the output file
251 0381 2 OUTPUT_FILE_NAME_DESC : BLOCK [8, BYTE], ! Name of output file
252 0382 2 OUTPUT_RESULT_NAME : BLOCK [NAM$C_MAXRSS, BYTE]; ! Place to store output file name
253 0383 2
254 0384 2
255 0385 2 OUTPUT_FILE_NAME_DESC [DSC$W_LENGTH] = %CHARCOUNT ('VMSMSG');
256 0386 2 OUTPUT_FILE_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
257 0387 2 OUTPUT_FILE_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
258 0388 2 OUTPUT_FILE_NAME_DESC [DSC$A_POINTER] = UPLIT ('VMSMSG');
259 0389 2
260 0390 2 Initialize the FAB, NAM and RAB for the output file
261 0391 2
262 P 0392 2 $FAB_INIT (FAB = OUTPUT_FAB, !
```



```
263 P 0393 FAC = (PUT),
264 P 0394 FOP = (OFP, SQO, DFW),
265 P 0395 ORG = SEQ,
266 P 0396 SHR = NIL,
267 P 0397 MRS = 132,
268 P 0398 RAT = CR,
269 P 0399 RFM = VAR,
270 P 0400 FNA = .OUTPUT_FILE_NAME_DESC [DSC$A_POINTER],
271 P 0401 FNS = .OUTPUT_FILE_NAME_DESC [DSC$W_LENGTH],
272 P 0402 DNA = UPLIT ('EDT$SRC:.TMP'),
273 P 0403 DNS = %CHARCOUNT ('EDT$SRC:.TMP'),
274 P 0404 NAM = OUTPUT_NAM);
275 P 0405 $NAM_INIT (NAM = OUTPUT_NAM,
276 P 0406 RSA = OUTPUT_RESULT_NAME,
277 P 0407 RSS = NAM$C_MAXRSS);
278 P 0408 $RAB_INIT (RAB = OUTPUT_RAB,
279 P 0409 RAC = SEQ,
280 P 0410 ROP = WBH,
281 P 0411 USZ = 132,
282 P 0412 UBF = OUTPUT_BUFFER,
283 P 0413 FAB = OUTPUT_FAB);
284 P 0414
285 P 0415 + Create the output file, and do the $CONNECT.
286 P 0416 -
287 P 0417 BEGIN
288 P 0418
289 P 0419 LOCAL
290 P 0420 CREATE_STATUS,
291 P 0421 CONNECT_STATUS;
292 P 0422
293 P 0423 CREATE_STATUS = $CREATE (FAB = OUTPUT_FAB);
294 P 0424
295 P 0425 IF ( NOT .CREATE_STATUS)
296 P 0426 THEN
297 P 0427 SIGNAL STOP (
298 P 0428 SHR$OPENOUT + (EDT$K_FAC_NO*65536) + ST$K_SEVERE,
299 P 0429 1,
300 P 0430 OUTPUT_FILE_NAME_DESC,
301 P 0431 .OUTPUT_FAB [FAB$L_ST$], .OUTPUT_FAB [FAB$L_STV]);
302 P 0432
303 P 0433 CONNECT_STATUS = $CONNECT (RAB = OUTPUT_RAB);
304 P 0434
305 P 0435 IF ( NOT .CONNECT_STATUS)
306 P 0436 THEN
307 P 0437 SIGNAL STOP (
308 P 0438 SHR$OPENOUT + (EDT$K_FAC_NO*65536) + ST$K_SEVERE,
309 P 0439 1,
310 P 0440 OUTPUT_FILE_NAME_DESC,
311 P 0441 .OUTPUT_RAB [RAB$L_ST$], .OUTPUT_RAB [RAB$L_STV]);
312 P 0442
313 P 0443 END;
314 P 0444 + Point the file name descriptor to the resultant name string.
315 P 0445 -
316 P 0446 OUTPUT_FILE_NAME_DESC [DSC$W_LENGTH] = .OUTPUT_NAM [NAM$B_RSL];
317 P 0447 OUTPUT_FILE_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
318 P 0448 OUTPUT_FILE_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
319 P 0449
```

```
0450 OUTPUT_FILE_NAME_DESC [DSC$A_POINTER] = .OUTPUT_NAM [NAM$L_RSA];
0451
0452 IF ( NOT WRITE_FILE (OUTPUT_RAB))
0453 THEN
0454     SIGNAL_STOP (
0455         SHR$WRITEERR + (EDT$K_FAC_NO*65536) + STS$K_SEVERE,
0456         1,
0457         OUTPUT_FILE_NAME_DESC,
0458         .OUTPUT_RAB[RAB$L_ST$], .OUTPUT_RAB[RAB$L_STV]);
0459
0460 BEGIN
0461     +
0462     - Close the output file.
0463
0464     LOCAL
0465     CLOSE_STATUS;
0466
0467     CLOSE_STATUS = $CLOSE (FAB = OUTPUT_FAB);
0468
0469     IF ( NOT .CLOSE_STATUS)
0470     THEN
0471         SIGNAL_STOP (
0472             SHR$CLOSEOUT + (EDT$K_FAC_NO*65536) + STS$K_SEVERE,
0473             1,
0474             OUTPUT_FILE_NAME_DESC,
0475             .OUTPUT_FAB[FAB$L_ST$], .OUTPUT_FAB[FAB$L_STV]);
0476
0477     END;
0478     RETURN (SS$NORMAL);
0479     END;
0480 1
```

! End of routine EDT\$WRIEDTMSG

```
.TITLE EDT$WRIEDTMSG EDT$WRIEDTMSG - write VMSMSG.MSG
.IDENT \V04-000\
```

```
.PSECT _EDT$CODE,NOWRT, SHR, PIC,2
```

```
00 50 4D 54 00 00 47 53 4D 53 4D 56 00000 P.AAA: .ASCII \VMSMSG\<0><0>
00 50 4D 54 2E 3A 43 52 53 54 44 45 00008 P.AAB: .ASCII \EDT$SRC:.TMP\<0>
```

```
.EXTRN STR$COPY_DX, STR$CONCAT
.EXTRN LIB$GET_INPUT, STR$COPY_R
.EXTRN STR$FREE1_DX, EDT$MSGTXT
.EXTRN SYSS$CREATE, SYSS$CONNECT
.EXTRN SYSS$CLOSE
```

007C 00000 EDT\$WRIEDTMSG:

```
.WORD Save R2,R3,R4,R5,R6
MOVAB LIB$STOP, R6
MOVAB -640(SP), SP
MOVL #17694726, OUTPUT_FILE_NAME_DESC
MOVAB P.AAA, OUTPUT_FILE_NAME_DESC+4
MOVCS #0, (SP), #0, #80, $RMS_PTR
MOVW #20483, $RMS_PTR
MOVL #536871008, $RMS_PTR+4
```

0050 8F

00

```
56 00000000G 00 9E 00002
5E FD80 CE 9E 00009
0100 CE 010E0006 8F D0 0000E
0104 CE D2 AF 9E 00017
6E 00 2C 0001D
FF2C CD FF2C CD 00024
FF30 CD 5003 8F B0 00027
CD 20000060 8F D0 0002E
```

0337

0385

0388

0404

0060	8F	00	FF42	CD	2001	8F	80	00037	MOVW	#8193, \$RMS_PTR+22	
			FF49	CD	0200	8F	80	0003E	MOVW	#512, \$RMS_PTR+29	
			FF4B	CD		02	90	00045	MOVB	#2, \$RMS_PTR+31	
			FF54	CD	FECC	CD	9E	0004A	MOVAB	OUTPUT_NAME, \$RMS_PTR+40	
			FF58	CD	0104	CE	DD	00051	MOVL	OUTPUT_FILE_NAME_DESC+4, \$RMS_PTR+44	
			FF5C	CD	99	AF	9E	00058	MOVAB	P.AAB, \$RMS_PTR+48	
			FF60	CD	0100	CE	90	0005E	MOVB	OUTPUT_FILE_NAME_DESC, \$RMS_PTR+52	
			FF61	CD		0B	90	00065	MOVB	#11, \$RMS_PTR+53	
			FF62	CD	84	8F	9B	0006A	MOVZBW	#132, \$RMS_PTR+54	
				6E		00	2C	00070	MOVC5	#0, (SP), #0, #96, \$RMS_PTR	0407
					FECC	CD		00077			
			FECC	CD	6002	8F	80	0007A	MOVW	#24578, \$RMS_PTR	
			FECE	CD		01	8E	00081	MNEGB	#1, \$RMS_PTR+2	
0044	8F	00	FED0	CD		6E	9E	00086	MOVAB	OUTPUT_RESULT_NAME, \$RMS_PTR+4	
				6E		00	2C	0008B	MOVC5	#0, (SP), #0, #68, \$RMS_PTR	0413
					0108	CE		00092			
			0108	CE	4401	8F	80	00095	MOVW	#17409, \$RMS_PTR	
			010C	CE	0400	8F	3C	0009C	MOVZWL	#1024, \$RMS_PTR+4	
					0126	CE	94	000A3	CLRB	\$RMS_PTR+30	
			0128	CE	84	8F	9B	000A7	MOVZBW	#132, \$RMS_PTR+32	
			012C	CE	FF7C	CD	9E	000AD	MOVAB	OUTPUT_BUFFER, \$RMS_PTR+36	
			FEC4	CD	FF2C	CD	9E	000B4	MOVAB	OUTPUT_FAB, \$RMS_PTR+60	
					FF2C	CD	9F	000BB	PUSHAB	OUTPUT_FAB	0423
00000000G		00				01	FB	000BF	CALLS	#1, SYS\$CREATE	
		14				50	E8	000C6	BLBS	CREATE_STATUS, 1\$	0425
		7E			FF34	CD	7D	000C9	MOVQ	OUTPUT_FAB+8, -(SP)	0431
					0108	CE	9F	000CE	PUSHAB	OUTPUT_FILE_NAME_DESC	0427
						01	DD	000D2	PUSHL	#1	
					008510A4	8F	DD	000D4	PUSHL	#8720548	0428
		66				05	FB	000DA	CALLS	#5, LIB\$STOP	
					0108	CE	9F	000DD	PUSHAB	OUTPUT_RAB	0433
00000000G		00				01	FB	000E1	CALLS	#1, SYS\$CONNECT	
		17				50	E8	000E8	BLBS	CONNECT_STATUS, 2\$	0435
					0114	CE	DD	000EB	PUSHL	OUTPUT_RAB+12	0441
					0114	CE	DD	000EF	PUSHL	OUTPUT_RAB+8	
					0108	CE	9F	000F3	PUSHAB	OUTPUT_FILE_NAME_DESC	0437
						01	DD	000F7	PUSHL	#1	
					00851CA4	8F	DD	000F9	PUSHL	#8720548	0438
		66				05	FB	000FF	CALLS	#5, LIB\$STOP	
			0100	CE	FECF	CD	9B	00102	MOVZBW	OUTPUT_NAM+3, OUTPUT_FILE_NAME_DESC	0447
			0102	CE	010E	8F	80	00109	MOVW	#270, OUTPUT_FILE_NAME_DESC+2	0448
			0104	CE	FED0	CD	DD	00110	MOVL	OUTPUT_NAM+4, OUTPUT_FILE_NAME_DESC+4	0450
					0108	CE	9F	00117	PUSHAB	OUTPUT_RAB	0452
0000V		CF				01	FB	0011B	CALLS	#1, WRITE_FILE	
		17				50	E8	00120	BLBS	R0, 3\$	
					0114	CE	DD	00123	PUSHL	OUTPUT_RAB+12	0458
					0114	CE	DD	00127	PUSHL	OUTPUT_RAB+8	
					0108	CE	9F	0012B	PUSHAB	OUTPUT_FILE_NAME_DESC	0454
						01	DD	0012F	PUSHL	#1	
					008510D4	8F	DD	00131	PUSHL	#8720596	0455
		66				05	FB	00137	CALLS	#5, LIB\$STOP	
					FF2C	CD	9F	0013A	PUSHAB	OUTPUT_FAB	0468
00000000G		00				01	FB	0013E	CALLS	#1, SYS\$CLOSE	
		14				50	E8	00145	BLBS	CLOSE_STATUS, 4\$	0470
		7E			FF34	CD	7D	00148	MOVQ	OUTPUT_FAB+8, -(SP)	0476
					0108	CE	9F	0014D	PUSHAB	OUTPUT_FILE_NAME_DESC	0472
						01	DD	00151	PUSHL	#1	

EDT\$WRIEDTMSG
V04-000

EDT\$WRIEDTMSG - write VMSMSG.MSG
EDT\$WRIEDTMSG - write VMSMSG.TMP

J 3
16-Sep-1984 02:18:31
14-Sep-1984 12:25:55

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]WRIEDTMSG.B32;1

Page 10
(4)

0085105C 8F DD 00153 PUSHL #8720476
66 05 FB 00159 CALLS #5, LIB\$STOP
50 01 D0 0015C 4\$: MOVL #1, R0
04 0015F RET

: 0473
: 0479
: 0480

; Routine Size: 352 bytes, Routine Base: _EDT\$CODE + 0014

EDT\$
V04-

; Rc

```
352 0481 1 XSBTTL 'WRITE_FILE - Actually write the file'
353 0482 1 ROUTINE WRITE_FILE (
354 0483 1     OUTPUT_RAB
355 0484 1 ) =
356 0485 1
357 0486 1
358 0487 1 ++
359 0488 1 FUNCTIONAL DESCRIPTION:
360 0489 1     This routine writes each record on the specified RAB.
361 0490 1
362 0491 1 CALLING SEQUENCE:
363 0492 1
364 0493 1     ret_status.wlc.v = WRITE_FILE (OUTPUT_RAB.mz.r)
365 0494 1
366 0495 1 FORMAL PARAMETERS:
367 0496 1
368 0497 1     OUTPUT_RAB          RAB onto which to write the text
369 0498 1
370 0499 1 IMPLICIT INPUTS:
371 0500 1
372 0501 1     NONE
373 0502 1
374 0503 1 IMPLICIT OUTPUTS:
375 0504 1
376 0505 1     None
377 0506 1
378 0507 1 COMPLETION STATUS:
379 0508 1
380 0509 1     $$$_NORMAL          Normal successful completion
381 0510 1     Any errors from RMS $PUT
382 0511 1
383 0512 1 SIDE EFFECTS:
384 0513 1
385 0514 1     Writes on the file connected to OUTPUT_RAB
386 0515 1
387 0516 1 --
388 0517 1
389 0518 1 BEGIN
390 0519 1
391 0520 1 MAP
392 0521 1     OUTPUT_RAB : REF $RAB_DECL;
393 0522 1
394 0523 1 LOCAL
395 0524 1
396 0525 1 ++
397 0526 1 Stuff for BUILD_TEXT_LINE
398 0527 1
399 0528 1     CTL_STR_DSC : BLOCK [8, BYTE],
400 0529 1     TEMP_STR_DSC : BLOCK [8, BYTE],
401 0530 1     TEMP_STRING : VECTOR [132, BYTE],
402 0531 1     OUT_LENGTH,
403 0532 1
404 0533 1 ++
405 0534 1 Stuff for PRINT_LINE
406 0535 1
407 0536 1     LINE_DESC : BLOCK [8, BYTE],
408 0537 1
409 0538 1 ++
410 0539 1 End of stuff for PRINT_LINE
```

```
409 0538 PRINTABLE_DESC : BLOCK [8, BYTE],
410 0539 HEX_DESC : BLOCK [8, BYTE],
411 0540 TEXT_DESC : BLOCK [8, BYTE],
412 0541 NAME_DESC : BLOCK [8, BYTE],
413 0542 SEVERITY_DESC : BLOCK [8, BYTE],
414 0543 NAME_LENGTH,
415 0544 TEXT_LENGTH,
416 0545 SEVERITY_LENGTH,
417 0546 SEVERITY_ADDR : REF VECTOR [, BYTE];
418 0547
419 0548
420 0549 + Set up TEMP_STR_DSC for BUILD_TEXT_LINE
421 0550 -
422 0551 TEMP_STR_DSC [DSC$W_LENGTH] = 132;
423 0552 TEMP_STR_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
424 0553 TEMP_STR_DSC [DSC$B_CLASS] = DSC$K_CLASS_S;
425 0554 TEMP_STR_DSC [DSC$A_POINTER] = CH$PTR (TEMP_STRING);
426 0555
427 0556 + Set up LINE_DESC for PRINT_LINE, etc.
428 0557 -
429 0558 INIT_DESCRIPTOR (LINE_DESC);
430 0559 INIT_DESCRIPTOR (PRINTABLE_DESC);
431 0560 INIT_DESCRIPTOR (HEX_DESC);
432 0561 INIT_DESCRIPTOR (TEXT_DESC);
433 0562 INIT_DESCRIPTOR (NAME_DESC);
434 0563 INIT_DESCRIPTOR (SEVERITY_DESC);
435 0564
436 0565 + Put out the initial information.
437 0566 -
438 0567 PRINT_LINE (<'!!! This file, VMSMSG.TMP, contains the definitions of the EDT'>, <' '>);
439 0568 PRINT_LINE (<'!!! messages for VAX/VMS. This file is read by the MESSAGE compiler'>, <' '>);
440 0569 PRINT_LINE (<'!!! to build an object file containing the EDT messages.'>, <' '>);
441 0570 PRINT_LINE (<'!!!'>, <' '>);
442 0571 PRINT_LINE (<' .TITLE EDT$VMSMSG EDT's message text'>, <' '>);
443 0572 PRINT_LINE (<'!!!'>, <' '>);
444 0573 PRINT_LINE (<' .FACILITY/SYSTEM EDT, !SL'>, <EDT$K_FAC_NO>);
445 0574
446 0575 + Write a line for each message
447 0576 -
448 0577
449 0578 INCR CODE FROM 0 TO 4095 DO
450 0579 BEGIN
451 0580 EDT$MSGTXT (CODE, SEVERITY_DESC, NAME_LENGTH, NAME_DESC, TEXT_LENGTH, TEXT_DESC);
452 0581
453 0582 + If the severity field is blank we are done.
454 0583 -
455 0584 SEVERITY_ADDR = .SEVERITY_DESC [DSC$A_POINTER];
456 0585
457 0586 IF (.SEVERITY_ADDR [0] EQL ' ') THEN EXITLOOP;
458 0587
459 0588 PRINT_LINE (<'!AS/!AS <!AS>'>, <NAME_DESC, SEVERITY_DESC, TEXT_DESC>);
460 0589 END;
461 0590
462 0591 + Write out the trailer line
463 0592 -
464 0593
465 0594 PRINT_LINE (<' .END'>, <' '>);
```



```

: 466      0595  2  !+
: 467      0596  2  !- All done.
: 468      0597  2  !-
: 469      0598  2  DISCARD_DESCRIPTOR (LINE_DESC);
: 470      0599  2  DISCARD_DESCRIPTOR (PRINTABLE_DESC);
: 471      0600  2  DISCARD_DESCRIPTOR (HEX_DESC);
: 472      0601  2  DISCARD_DESCRIPTOR (TEXT_DESC);
: 473      0602  2  DISCARD_DESCRIPTOR (NAME_DESC);
: 474      0603  2  DISCARD_DESCRIPTOR (SEVERITY_DESC);
: 475      0604  2  RETURN (SS$NORMAL);
: 476      0605  1  END;
                                ! End of routine WRITE_FILE
```

```

56 20 2C 65 6C 69 66 20 73 69 68 54 20 21 21 00174 P.AAC: .ASCII \!! This file, VMSMSG.TMP, contains the d\
74 6E 6F 63 20 2C 50 4D 54 2E 47 53 4D 53 4D 00183
64 20 65 68 74 20 73 6E 69 61 00192
74 20 66 6F 20 73 6E 6F 69 74 69 6E 69 66 65 0019C .ASCII \efinitions of the EDT\<0><0><0>
00 00 00 54 44 45 20 65 68 001AB
72 6F 66 20 73 65 67 61 73 73 65 6D 20 21 21 001B4 P.AAD: .ASCII \!! messages for VAX/VMS. This file is r\
73 69 68 54 20 20 2E 53 4D 56 2F 58 41 56 20 001C3
72 20 73 69 20 65 6C 69 66 20 001D2
53 53 45 4D 20 65 68 74 20 79 62 20 64 61 65 001DC .ASCII \e ad by the MESSAGE compiler\<0>
00 72 65 6C 69 70 6D 6F 63 20 45 47 41 001EB
20 6E 61 20 64 6C 69 75 62 20 6F 74 20 21 21 001F8 P.AAE: .ASCII \!! to build an object file containing th\
6E 6F 63 20 65 6C 69 66 20 74 63 65 6A 62 6F 00207
68 74 20 67 6E 69 6E 69 61 74 00216
2E 73 65 67 61 73 73 65 6D 20 54 44 45 20 65 00220 .ASCII \e EDT messages.\<0>
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0022F
20 45 4C 54 49 54 2E 20 20 20 20 20 20 20 20 00230 P.AAF: .ASCII \!!\<0><0>
27 54 44 45 20 47 53 4D 53 4D 56 24 54 44 45 00234 P.AAG: .ASCII \ .TITLE EDT$VMSMSG EDT's message \
20 65 67 61 73 73 65 6D 20 73 00243
74 78 65 74 0025C .ASCII \text\
00 00 21 21 00260 P.AAH: .ASCII \!!\<0><0>
45 54 53 59 53 2F 59 54 49 4C 49 43 41 46 2E 00264 P.AAI: .ASCII \.FACILITY/SYSTEM EDT, !SL\<0><0><0>
00 00 00 00 4C 53 21 20 2C 54 44 45 20 4D 00273
00 00 3E 53 41 21 3C 20 53 41 21 2F 53 41 21 00280 P.AAJ: .ASCII \!AS/!AS <!AS>\<0><0><0>
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0028F
44 4E 45 2E 00290 P.AAK: .ASCII \.END\
                                .EXTRN SYS$FAO
```

01FC 0000 WRITE_FILE:

```

58 0000V CF 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8 : 0482
57 00000000G 00 9E 00007 MOVAB PRINT, R8
56 00000000G 00 9E 0000E MOVAB STR$FREE1 DX, R7
55 00000000G 00 9E 00015 MOVAB STR$COPY_R, R6
54 00000000G 00 9E 0001C MOVAB SYS$FAO, R5
5E FF2C CE 9E 00023 MOVAB LIB$STOP, R4
F0 AD 010E0084 8F D0 00028 MOVAB -212(SP), SP : 0551
F4 AD 40 AE 9E 00030 MOVAB #17694852, TEMP_STR_DSC : 0554
38 AE 020E0000 8F D0 00035 MOVAB TEMP_STRING, TEMP_STR_DSC+4 : 0558
3C AE D4 0003D CLRL LINE_DESC+4
30 AE 020E0000 8F D0 00040 MOVAB #34471936, PRINTABLE_DESC : 0559
34 AE D4 00048 CLRL PRINTABLE_DESC+4
```

28	AE	020E0000	8F	DO	0004B	MOVL	#34471936, HEX_DESC	0560
		2C	AE	D4	00053	CLRL	HEX_DESC+4	
20	AE	020E0000	8F	DO	00056	MOVL	#34471936, TEXT_DESC	0561
		24	AE	D4	0005E	CLRL	TEXT_DESC+4	
18	AE	020E0000	8F	DO	00061	MOVL	#34471936, NAME_DESC	0562
		1C	AE	D4	00069	CLRL	NAME_DESC+4	
10	AE	020E0000	8F	DO	0006C	MOVL	#34471936, SEVERITY_DESC	0563
		14	AE	D4	00074	CLRL	SEVERITY_DESC+4	
F8	AD	010E003D	8F	DO	00077	MOVL	#17694787, CTL_STR_DSC	0567
FC	AD	FE5D	CF	9E	0007F	MOVAB	P.AAC, CTL_STR_DSC+4	
			20	DD	00085	PUSHL	#32	
		F0	AD	9F	00087	PUSHAB	TEMP_STR_DSC	
		14	AE	9F	0008A	PUSHAB	OUT_LENGTH	
		F8	AD	9F	0008D	PUSHAB	CTL_STR_DSC	
65			04	FB	00090	CALLS	#4, SYS\$FAO	
05			50	E8	00093	BLBS	FAO_STATUS, 1\$	
			50	DD	00096	PUSHL	FAO_STATUS	
64			01	FB	00098	CALLS	#1, LIB\$STOP	
		F4	AD	DD	0009B	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	0009E	PUSHAB	OUT_LENGTH	
		40	AE	9F	000A1	PUSHAB	LINE_DESC	
66			03	FB	000A4	CALLS	#3, STR\$COPY_R	
7D			50	E9	000A7	BLBC	BUILD_STATUS, 4\$	
		38	AE	9F	000AA	PUSHAB	LINE_DESC	
52		04	AC	DO	000AD	MOVL	OUTPUT_RAB, R2	
			52	DD	000B1	PUSHL	R2	
68			02	FB	000B3	CALLS	#2, PRINT	
79			50	E9	000B6	BLBC	PRINT_STATUS, 5\$	
F8	AD	010E0043	8F	DO	000B9	MOVL	#17694787, CTL_STR_DSC	0568
FC	AD	FE5B	CF	9E	000C1	MOVAB	P.AAD, CTL_STR_DSC+4	
			20	DD	000C7	PUSHL	#32	
		F0	AD	9F	000C9	PUSHAB	TEMP_STR_DSC	
		14	AE	9F	000CC	PUSHAB	OUT_LENGTH	
		F8	AD	9F	000CF	PUSHAB	CTL_STR_DSC	
65			04	FB	000D2	CALLS	#4, SYS\$FAO	
05			50	E8	000D5	BLBS	FAO_STATUS, 2\$	
			50	DD	000D8	PUSHL	FAO_STATUS	
64			01	FB	000DA	CALLS	#1, LIB\$STOP	
		F4	AD	DD	000DD	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	000E0	PUSHAB	OUT_LENGTH	
		40	AE	9F	000E3	PUSHAB	LINE_DESC	
66			03	FB	000E6	CALLS	#3, STR\$COPY_R	
79			50	E9	000E9	BLBC	BUILD_STATUS, 7\$	
		38	AE	9F	000EC	PUSHAB	LINE_DESC	
			52	DD	000EF	PUSHL	R2	
68			02	FB	000F1	CALLS	#2, PRINT	
79			50	E9	000F4	BLBC	PRINT_STATUS, 8\$	
F8	AD	010E0037	8F	DO	000F7	MOVL	#17694775, CTL_STR_DSC	0569
FC	AD	FE61	CF	9E	000FF	MOVAB	P.AAE, CTL_STR_DSC+4	
			20	DD	00105	PUSHL	#32	
		F0	AD	9F	00107	PUSHAB	TEMP_STR_DSC	
		14	AE	9F	0010A	PUSHAB	OUT_LENGTH	
		F8	AD	9F	0010D	PUSHAB	CTL_STR_DSC	
65			04	FB	00110	CALLS	#4, SYS\$FAO	
05			50	E8	00113	BLBS	FAO_STATUS, 3\$	
			50	DD	00116	PUSHL	FAO_STATUS	
64			01	FB	00118	CALLS	#1, LIB\$STOP	

		F4	AD	DD	0011B	3\$:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	0011E		PUSHAB	OUT_LENGTH	
		40	AE	9F	00121		PUSHAB	LINE_DESC	
	66		03	FB	00124		CALLS	#3, STR\$COPY_R	
	79		50	E9	00127	4\$:	BLBC	BUILD_STATUS, 10\$	
		38	AE	9F	0012A		PUSHAB	LINE_DESC	
			52	DD	0012D		PUSHL	R2	
	68		02	FB	0012F		CALLS	#2, PRINT	
	79		50	E9	00132	5\$:	BLBC	PRINT_STATUS, 11\$	
FB	AD	010E0002	8F	DD	00135		MOVL	#17694722, CTL_STR_DSC	
FC	AD	FE5B	CF	9E	0013D		MOVAB	P.AAF, CTL_STR_DSC+4	0570
			20	DD	00143		PUSHL	#32	
		F0	AD	9F	00145		PUSHAB	TEMP_STR_DSC	
		14	AE	9F	00148		PUSHAB	OUT_LENGTH	
		F8	AD	9F	0014B		PUSHAB	CTL_STR_DSC	
	65		04	FB	0014E		CALLS	#4, -SYS\$FAO	
	05		50	E8	00151		BLBS	FAO_STATUS, 6\$	
			50	DD	00154		PUSHL	FAO_STATUS	
	64		01	FB	00156		CALLS	#1, -LIB\$STOP	
		F4	AD	DD	00159	6\$:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	0015C		PUSHAB	OUT_LENGTH	
		40	AE	9F	0015F		PUSHAB	LINE_DESC	
	66		03	FB	00162		CALLS	#3, STR\$COPY_R	
	79		50	E9	00165	7\$:	BLBC	BUILD_STATUS, 13\$	
		38	AE	9F	00168		PUSHAB	LINE_DESC	
			52	DD	0016B		PUSHL	R2	
	68		02	FB	0016D		CALLS	#2, PRINT	
	79		50	E9	00170	8\$:	BLBC	PRINT_STATUS, 14\$	
FB	AD	010E002C	8F	DD	00173		MOVL	#17694764, CTL_STR_DSC	
FC	AD	FE21	CF	9E	0017B		MOVAB	P.AAG, CTL_STR_DSC+4	0571
			20	DD	00181		PUSHL	#32	
		F0	AD	9F	00183		PUSHAB	TEMP_STR_DSC	
		14	AE	9F	00186		PUSHAB	OUT_LENGTH	
		F8	AD	9F	00189		PUSHAB	CTL_STR_DSC	
	65		04	FB	0018C		CALLS	#4, -SYS\$FAO	
	05		50	E8	0018F		BLBS	FAO_STATUS, 9\$	
			50	DD	00192		PUSHL	FAO_STATUS	
	64		01	FB	00194		CALLS	#1, -LIB\$STOP	
		F4	AD	DD	00197	9\$:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	0019A		PUSHAB	OUT_LENGTH	
		40	AE	9F	0019D		PUSHAB	LINE_DESC	
	66		03	FB	001A0		CALLS	#3, STR\$COPY_R	
	78		50	E9	001A3	10\$:	BLBC	BUILD_STATUS, 16\$	
		38	AE	9F	001A6		PUSHAB	LINE_DESC	
			52	DD	001A9		PUSHL	R2	
	68		02	FB	001AB		CALLS	#2, PRINT	
	78		50	E9	001AE	11\$:	BLBC	PRINT_STATUS, 17\$	
FB	AD	010E0002	8F	DD	001B1		MOVL	#17694722, CTL_STR_DSC	
FC	AD	FE0F	CF	9E	001B9		MOVAB	P.AAH, CTL_STR_DSC+4	0572
			20	DD	001BF		PUSHL	#32	
		F0	AD	9F	001C1		PUSHAB	TEMP_STR_DSC	
		14	AE	9F	001C4		PUSHAB	OUT_LENGTH	
		F8	AD	9F	001C7		PUSHAB	CTL_STR_DSC	
	65		04	FB	001CA		CALLS	#4, -SYS\$FAO	
	05		50	E8	001CD		BLBS	FAO_STATUS, 12\$	
			50	DD	001D0		PUSHL	FAO_STATUS	
	64		01	FB	001D2		CALLS	#1, -LIB\$STOP	

		F4	AD	DD	001D5	12%:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	001D8		PUSHAB	OUT_LENGTH	
		40	AE	9F	001DB		PUSHAB	LINE_DESC	
66			03	FB	001DE		CALLS	#3, STR\$COPY_R	
48			50	E9	001E1	13%:	BLBC	BUILD_STATUS, 17\$	
		38	AE	9F	001E4		PUSHAB	LINE_DESC	
			52	DD	001E7		PUSHL	R2	
68			02	FB	001E9		CALLS	#2, PRINT	
3D			50	E9	001EC	14%:	BLBC	PRINT_STATUS, 17\$	
F8	AD	010E0019	8F	DD	001EF		MOVL	#17694745, CTL_STR_DSC	0573
FC	AD	FDD5	CF	9E	001F7		MOVAB	P.AAJ, CTL_STR_DSC+4	
		7E	85	8F	9A		MOVZBL	#133, -(SPT	
			F0	AD	9F	00201	PUSHAB	TEMP_STR_DSC	
			14	AE	9F	00204	PUSHAB	OUT_LENGTH	
			F8	AD	9F	00207	PUSHAB	CTL_STR_DSC	
65			04	FB	0020A		CALLS	#4, SYS\$FAO	
05			50	E8	0020D		BLBS	FAO_STATUS, 15\$	
			50	DD	00210		PUSHL	FAO_STATUS	
64			01	FB	00212		CALLS	#1, LIB\$STOP	
		F4	AD	DD	00215	15%:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	00218		PUSHAB	OUT_LENGTH	
		40	AE	9F	0021B		PUSHAB	LINE_DESC	
66			03	FB	0021E		CALLS	#3, STR\$COPY_R	
72			50	E9	00221	16%:	BLBC	BUILD_STATUS, 20\$	
		38	AE	9F	00224		PUSHAB	LINE_DESC	
			52	DD	00227		PUSHL	R2	
68			02	FB	00229		CALLS	#2, PRINT	
67			50	E9	0022C	17%:	BLBC	PRINT_STATUS, 20\$	
		08	AE	D4	0022F		CLRL	CODE	0578
		20	AE	9F	00232	18%:	PUSHAB	TEXT_DESC	0580
		04	AE	9F	00235		PUSHAB	TEXT_LENGTH	
		20	AE	9F	00238		PUSHAB	NAME_DESC	
		10	AE	9F	0023B		PUSHAB	NAME_LENGTH	
		20	AE	9F	0023E		PUSHAB	SEVERITY_DESC	
		1C	AE	9F	00241		PUSHAB	CODE	
00000000G	00		06	FB	00244		CALLS	#6, EDT\$MSGTXT	
	53	14	AE	DD	0024B		MOVL	SEVERITY_DESC+4, SEVERITY_ADDR	0584
	20		63	91	0024F		CMPB	(SEVERITY_ADDR), #32	0586
			4E	13	00252		BEQL	21\$	
F8	AD	010E000D	8F	DD	00254		MOVL	#17694733, CTL_STR_DSC	0588
FC	AD	FD8C	CF	9E	0025C		MOVAB	P.AAJ, CTL_STR_DSC+4	
		20	AE	9F	00262		PUSHAB	TEXT_DESC	
		14	AE	9F	00265		PUSHAB	SEVERITY_DESC	
		20	AE	9F	00268		PUSHAB	NAME_DESC	
		F0	AD	9F	0026B		PUSHAB	TEMP_STR_DSC	
		1C	AE	9F	0026E		PUSHAB	OUT_LENGTH	
		F8	AD	9F	00271		PUSHAB	CTL_STR_DSC	
65			06	FB	00274		CALLS	#6, SYS\$FAO	
05			50	E8	00277		BLBS	FAO_STATUS, 19\$	
			50	DD	0027A		PUSHL	FAO_STATUS	
64			01	FB	0027C		CALLS	#1, LIB\$STOP	
		F4	AD	DD	0027F	19%:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	00282		PUSHAB	OUT_LENGTH	
		40	AE	9F	00285		PUSHAB	LINE_DESC	
66			03	FB	00288		CALLS	#3, STR\$COPY_R	
4F			50	E9	0028B		BLBC	BUILD_STATUS, 23\$	
		38	AE	9F	0028E		PUSHAB	LINE_DESC	

90	08	68	52	DD	00291	PUSHL	R2		
	F8	44	02	FB	00293	CALLS	#2, PRINT		
	FC	AD	50	E9	00296	20%:	BLBC	PRINT STATUS, 23%	
		AD	8F	F3	00299	AOBLEQ	#4095, CODE, 18%		0578
			8F	D0	002A2	21%:	MOVL	#17694724, CTL_STR_DSC	0594
			CF	9E	002AA	MOVAB	P_AAK, CTL_STR_DSC+4		
			20	DD	002B0	PUSHL	#32		
		F0	AD	9F	002B2	PUSHAB	TEMP_STR_DSC		
		14	AE	9F	002B5	PUSHAB	OUT_LENGTH		
		F8	AD	9F	002B8	PUSHAB	CTL_STR_DSC		
		65	04	FB	002BB	CALLS	#4, SYS\$FAO		
		05	50	E8	002BE	BLBS	FAO STATUS, 22%		
		64	50	DD	002C1	PUSHL	FAO STATUS		
			01	FB	002C3	CALLS	#1, LIB\$STOP		
		F4	AD	DD	002C6	22%:	PUSHL	TEMP_STR_DSC+4	
		10	AE	9F	002C9	PUSHAB	OUT_LENGTH		
		40	AE	9F	002CC	PUSHAB	LINE_DESC		
		66	03	FB	002CF	CALLS	#3, STR\$COPY_R		
		62	50	E9	002D2	BLBC	BUILD STATUS, 30%		
		38	AE	9F	002D5	PUSHAB	LINE_DESC		
			52	DD	002D8	PUSHL	R2		
		68	02	FB	002DA	CALLS	#2, PRINT		
		57	50	E9	002DD	23%:	BLBC	PRINT STATUS, 30%	
			38	AE	9F	002E0	PUSHAB	LINE_DESC	0598
		67	01	FB	002E3	CALLS	#1, STR\$FREE1 DX		
		05	50	E8	002E6	BLBS	FREE STATUS, 24%		
		64	50	DD	002E9	PUSHL	FREE STATUS		
			01	FB	002EB	CALLS	#1, LIB\$STOP		
		67	30	AE	9F	002EE	24%:	PUSHAB	PRINTABLE_DESC
		05	01	FB	002F1	CALLS	#1, STR\$FREE1 DX		0599
			50	E8	002F4	BLBS	FREE STATUS, 25%		
		64	50	DD	002F7	PUSHL	FREE STATUS		
			01	FB	002F9	CALLS	#1, LIB\$STOP		
		67	28	AE	9F	002FC	25%:	PUSHAB	HEX_DESC
		05	01	FB	002FF	CALLS	#1, STR\$FREE1 DX		0600
			50	E8	00302	BLBS	FREE STATUS, 26%		
		64	50	DD	00305	PUSHL	FREE STATUS		
			01	FB	00307	CALLS	#1, LIB\$STOP		
		67	20	AE	9F	0030A	26%:	PUSHAB	TEXT_DESC
		05	01	FB	0030D	CALLS	#1, STR\$FREE1 DX		0601
			50	E8	00310	BLBS	FREE STATUS, 27%		
		64	50	DD	00313	PUSHL	FREE STATUS		
			01	FB	00315	CALLS	#1, LIB\$STOP		
		67	18	AE	9F	00318	27%:	PUSHAB	NAME_DESC
		05	01	FB	0031B	CALLS	#1, STR\$FREE1 DX		0602
			50	E8	0031E	BLBS	FREE STATUS, 28%		
		64	50	DD	00321	PUSHL	FREE STATUS		
			01	FB	00323	CALLS	#1, LIB\$STOP		
		67	10	AE	9F	00326	28%:	PUSHAB	SEVERITY_DESC
		05	01	FB	00329	CALLS	#1, STR\$FREE1 DX		0603
			50	E8	0032C	BLBS	FREE STATUS, 29%		
		64	50	DD	0032F	PUSHL	FREE STATUS		
			01	FB	00331	CALLS	#1, LIB\$STOP		
		50	01	D0	00334	29%:	MOVL	#1, R0	0604
			04	00337	30%:	RET			0605

EDTSURIEDTMSG
V04-000

EDTSURIEDTMSG - write VM\$MSG.MSG
WRITE_FILE - Actually write the file

E 4
16-Sep-1984 02:18:31
14-Sep-1984 12:25:55

VAX-11 B11ss-32 V4.0-742
[EDT.SRC]WRIEDTMSG.B32;1

Page 18
(5)

EDT
V04-


```
478 0606 1 XSBTTL 'PRINT - print a text line on a file'
479 0607 1 ROUTINE PRINT (
480 0608 1     RAB_ADDR
481 0609 1     TEXT_LINE
482 0610 1 ) =
483 0611 1
484 0612 1
485 0613 1 ++
486 0614 1 FUNCTIONAL DESCRIPTION:
487 0615 1     This routine interfaces to RMS to print a line of text.
488 0616 1
489 0617 1 CALLING SEQUENCE:
490 0618 1
491 0619 1     ret_status.wlc.v = PRINT (RAB_ADDR.mz.r, TEXT_LINE.rt.dx)
492 0620 1
493 0621 1 FORMAL PARAMETERS:
494 0622 1
495 0623 1     RAB_ADDR           Pointer to the RAB onto which to print the line of text
496 0624 1     TEXT_LINE         Descriptor for the line of text to be printed.
497 0625 1
498 0626 1 IMPLICIT INPUTS:
499 0627 1
500 0628 1     NONE
501 0629 1
502 0630 1 IMPLICIT OUTPUTS:
503 0631 1
504 0632 1     NONE
505 0633 1
506 0634 1 COMPLETION STATUS:
507 0635 1
508 0636 1     $$$_NORMAL        Normal successful completion
509 0637 1     All RMS errors are returned to the caller, so that they can be
510 0638 1     signalled with the file name.
511 0639 1
512 0640 1 SIDE EFFECTS:
513 0641 1
514 0642 1     Does a $PUT to the RAB.
515 0643 1
516 0644 1 --
517 0645 1
518 0646 2 BEGIN
519 0647 2
520 0648 2 MAP
521 0649 2     RAB_ADDR : REF $RAB DECL,
522 0650 2     TEXT_LINE : REF BLOCK [8, BYTE];
523 0651 2
524 0652 2 LOCAL
525 0653 2     PUT_STATUS;
526 0654 2
527 0655 2
528 0656 2 ++ Fill in the RAB fields.
529 0657 2 --
530 0658 2     RAB_ADDR [RAB$W_RSZ] = .TEXT_LINE [DSC$W_LENGTH];
531 0659 2     RAB_ADDR [RAB$SL_RBF] = .TEXT_LINE [DSC$A_POINTER];
532 0660 2
533 0661 2 ++ Now do the $PUT
534 0662 2 --
```

; Routine Size: 35 bytes, Routine Base: _EDT\$CODE + 05CC

```
.. 542 0669 1 %SBTTL 'HEX_TEXT - Return a binary string in hexadecimal'
543 0670 1 ROUTINE HEX_TEXT (
544 0671 1     OUTPUT_DESC,
545 0672 1     INPUT_LEN,
546 0673 1     INPUT_ADDR
547 0674 1 ) =
548 0675 1
549 0676 1
550 0677 1 ++
551 0678 1 FUNCTIONAL DESCRIPTION:
552 0679 1     This routine converts an arbitrary string of bytes into hex, so it
553 0680 1     can be printed. Early bytes are put to the right of later bytes.
554 0681 1
555 0682 1 CALLING SEQUENCE:
556 0683 1
557 0684 1     status.wlc.v = HEX_TEXT (OUTPUT_desc.wt.dx, INPUT_LEN.rl.v, INPUT_ADDR.ra.v)
558 0685 1
559 0686 1 FORMAL PARAMETERS:
560 0687 1
561 0688 1     output_desc    Where the result text is stored.
562 0689 1     input_len      Number of bytes of input
563 0690 1     input_addr     Address of first input byte
564 0691 1
565 0692 1 IMPLICIT INPUTS:
566 0693 1
567 0694 1     NONE
568 0695 1
569 0696 1 IMPLICIT OUTPUTS:
570 0697 1
571 0698 1     NONE
572 0699 1
573 0700 1 COMPLETION STATUS:
574 0701 1
575 0702 1     $$$_NORMAL      Normal successful completion
576 0703 1     Any errors from STR$CONCAT
577 0704 1     Any errors from STR$COPY_DX
578 0705 1
579 0706 1 SIDE EFFECTS:
580 0707 1
581 0708 1     Calls STR$CONCAT and STR$COPY_DX, thus manipulating string storage.
582 0709 1
583 0710 1 --
584 0711 1
585 0712 2 BEGIN
586 0713 2
587 0714 2 MAP
588 0715 2     INPUT_ADDR : REF VECTOR [ , BYTE]
589 0716 2     OUTPUT_DESC : REF BLOCK [8, BYTE];
590 0717 2
591 0718 2 LOCAL
592 0719 2     INTER_DESC : BLOCK [8, BYTE],
593 0720 2     DIGIT_DESC : BLOCK [8, BYTE],
594 0721 2     DIGIT,
595 0722 2     STATUS;
596 0723 2
597 0724 2 INIT_DESCRIPTOR (INTER_DESC);
598 0725 2     DIGIT_DESC [DSC$W_LENGTH] = 1;
```



```
599 0726 2 DIGIT_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
600 0727 DIGIT_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
601 0728 DIGIT_DESC [DSC$A_POINTER] = DIGIT;
602 0729
603 0730 INCR CHAR_NO FROM 1 TO .INPUT_LEN DO
604 0731 BEGIN
605 0732
606 0733 LOCAL
607 0734 CHAR;
608 0735
609 0736 CHAR = .INPUT_ADDR [.CHAR_NO - 1];
610 0737 DIGIT = (.CHAR AND 15) + '0';
611 0738
612 0739 IF (.DIGIT GTR '9') THEN DIGIT = .DIGIT - 10 - '0' + 'A';
613 0740
614 0741 STATUS = STR$CONCAT (INTER_DESC, DIGIT_DESC, INTER_DESC);
615 0742
616 0743 IF ( NOT .STATUS) THEN RETURN (.STATUS);
617 0744
618 0745 DIGIT = (.CHAR^4) + '0';
619 0746
620 0747 IF (.DIGIT GTR '9') THEN DIGIT = .DIGIT - 10 - '0' + 'A';
621 0748
622 0749 STATUS = STR$CONCAT (INTER_DESC, DIGIT_DESC, INTER_DESC);
623 0750
624 0751 IF ( NOT .STATUS) THEN RETURN (.STATUS);
625 0752
626 0753 END;
627 0754
628 0755 STATUS = STR$COPY DX (.OUTPUT_DESC, INTER_DESC);
629 0756 DISCARD_DESCRIPTOR (INTER_DESC);
630 0757 RETURN (.STATUS);
631 0758 1
```

! End of routine HEX_TEXT

003C 00000 HEX_TEXT:

		55	00000000G	00	9E	00002	WORD	Save R2,R3,R4,R5	0670
		5E		14	C2	00009	MOVAB	STR\$CONCAT, R5	
		OC	AE 020E0000	8F	D0	0000C	SUBL2	#20, SP	
			10	AE	D4	00014	MOVL	#34471936, INTER_DESC	0724
		04	AE 010E0001	8F	D0	00017	CLRL	INTER_DESC+4	
		08	AE	6E	9E	0001F	MOVL	#17694721, DIGIT_DESC	0725
				53	D4	00023	MOVAB	DIGIT, DIGIT_DESC+4	0728
				4E	11	00025	CLRL	CHAR_NO	0736
							BRB	48	
	50	53	OC	AC	C1	00027	ADDL3	INPUT_ADDR, CHAR_NO, R0	
		52	FF	A0	9A	0002C	MOVZBL	-1(R0), CHAR	
6E	52	04		00	EF	00030	EXTZV	#0, #4, CHAR, DIGIT	0737
		6E		30	C0	00035	ADDL2	#48, DIGIT	
		39		6E	D1	00038	CMPL	DIGIT, #57	0739
				03	15	0003B	BLEQ	28	
		6E		07	C0	0003D	ADDL2	#7, DIGIT	
			OC	AE	9F	00040	PUSHAB	INTER_DESC	0741
			08	AE	9F	00043	PUSHAB	DIGIT_DESC	

		14	AE 9F 00046	PUSHAB	INTER_DESC	
	65		03 FB 00049	CALLS	#3, STR\$CONCAT	
	54		50 D0 0004C	MOVL	R0, STATUS	
	4E		54 E9 0004F	BLBC	STATUS, 5\$	0743
52	52	FC	8F 78 00052	ASHL	#-4, CHAR, R2	0745
	6E	30	A2 9E 00057	MOVAB	48(R2), DIGIT	
	39		6E D1 0005B	CML	DIGIT, #57	0747
			03 15 0005E	BLEQ	3\$	
	6E		07 C0 00060	ADDL2	#7, DIGIT	
		0C	AE 9F 00063	PUSHAB	INTER_DESC	0749
		08	AE 9F 00066	PUSHAB	DIGIT_DESC	
		14	AE 9F 00069	PUSHAB	INTER_DESC	
	65		03 FB 0006C	CALLS	#3, STR\$CONCAT	
	54		50 D0 0006F	MOVL	R0, STATUS	
	2B		54 E9 00072	BLBC	STATUS, 5\$	0751
AD	53	08	AC F3 00075	AOBLEQ	INPUT_LEN, CHAR_NO, 1\$	0750
		0C	AE 9F 0007A	PUSHAB	INTER_DESC	0755
		04	AC DD 0007D	PUSHL	OUTPUT_DESC	
00000000G	00		02 FB 00080	CALLS	#2, STR\$COPY_DX	
	54		50 D0 00087	MOVL	R0, STATUS	
		0C	AE 9F 0008A	PUSHAB	INTER_DESC	0756
00000000G	00		01 FB 0008D	CALLS	#1, STR\$FREE1_DX	
	09		50 E8 00094	BLBS	FREE_STATUS, 5\$	
			50 DD 00097	PUSHL	FREE_STATUS	
00000000G	00		01 FB 00099	CALLS	#1, LIB\$STOP	
	50		54 D0 000A0	MOVL	STATUS, R0	0757
			04 000A3	RET		0758

; Routine Size: 164 bytes, Routine Base: _EDT\$CODE + 05EF

```
0759 1 $SBTTL 'PRINTABLE_TEXT - Return a binary string in ASCII, printable'
0760 1 ROUTINE PRINTABLE_TEXT (
0761 1     OUTPUT_DESC,      Return a binary string in printable ASCII
0762 1     INPUT_LEN,        Descriptor to receive the text
0763 1     INPUT_ADDR,       Number of input bytes
0764 1     ) =              Address of start of input
0765 1
0766 1
0767 1 ++
0768 1 FUNCTIONAL DESCRIPTION:
0769 1     This routine converts an arbitrary string of bytes into ASCII, representing
0770 1     unprintable characters in hexadecimal so the result can be printed.
0771 1
0772 1 CALLING SEQUENCE:
0773 1
0774 1     status.wlc.v = PRINTABLE_TEXT (OUTPUT_desc.wt.dx, INPUT_LEN.rl.v, INPUT_ADDR.ra.v)
0775 1
0776 1 FORMAL PARAMETERS:
0777 1
0778 1     OUTPUT_DESC      Where the result text is stored.
0779 1     INPUT_LEN        Number of bytes of input
0780 1     INPUT_ADDR       Address of first input byte
0781 1
0782 1 IMPLICIT INPUTS:
0783 1
0784 1     NONE
0785 1
0786 1 IMPLICIT OUTPUTS:
0787 1
0788 1     NONE
0789 1
0790 1 COMPLETION STATUS:
0791 1
0792 1     $$$_NORMAL      Normal successful completion
0793 1     Any errors from STR$CONCAT
0794 1     Any errors from STR$COPY_DX
0795 1
0796 1 SIDE EFFECTS:
0797 1
0798 1     Calls STR$CONCAT and STR$COPY_DX, thus manipulating string storage.
0799 1
0800 1 --
0801 1
0802 1 BEGIN
0803 1
0804 1 MAP
0805 1     INPUT_ADDR : REF VECTOR [ 8, BYTE],
0806 1     OUTPUT_DESC : REF BLOCK [8, BYTE];
0807 1
0808 1 LOCAL
0809 1     INTER_DESC : BLOCK [8, BYTE],
0810 1     CHAR_DESC  : BLOCK [8, BYTE],
0811 1     CHAR_REP   : VECTOR [4, BYTE],
0812 1     STATOS;
0813 1
0814 1 INIT_DESCRIPTOR (INTER_DESC);
0815 1 CHAR_DESC [DSC$W_LENGTH] = 1;
```



```
690 0816 2 CHAR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
691 0817 2 CHAR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
692 0818 2 CHAR_DESC [DSC$A_POINTER] = CHAR_REP [0];
693 0819 2
694 0820 2 INCR CHAR_NO FROM 1 TO .INPUT_LEN DO
695 0821 2 BEGIN
696 0822 2
697 0823 2 LOCAL
698 0824 2 CHAR;
699 0825 2
700 0826 2 CHAR = .INPUT_ADDR [.CHAR_NO - 1];
701 0827 2
702 0828 2 IF ((.CHAR GEQ 'X'20') AND
703 0829 2 (.CHAR LSS 'X'7F') AND
704 0830 2 (.CHAR NEQ '<') AND
705 0831 2 (.CHAR NEQ '''))
706 0832 2 THEN
707 0833 2 BEGIN
708 0834 2 + Show character as itself.
709 0835 2 -
710 0836 2 CHAR_REP [0] = .CHAR;
711 0837 2 CHAR_DESC [DSC$W_LENGTH] = 1;
712 0838 2 END
713 0839 2 ELSE
714 0840 2 BEGIN
715 0841 2 + The character is not printable. Represent it by <>. To avoid
716 0842 2 ambiguity, '<' and '' are also represented this way. Control characters
717 0843 2 SOH through SUB are represented by <^letter>; others characters by <hex>.
718 0844 2 -
719 0845 2 CHAR_REP [0] = '<';
720 0846 2
721 0847 2 IF (((.CHAR + 'X'40') GEQ 'A') AND ((.CHAR + 'X'40') LEQ 'Z'))
722 0848 2 THEN
723 0849 2 BEGIN
724 0850 2 CHAR_REP [1] = '^';
725 0851 2 CHAR_REP [2] = .CHAR + 'X'40';
726 0852 2 END
727 0853 2 ELSE
728 0854 2 BEGIN
729 0855 2 LOCAL
730 0856 2 DIGIT;
731 0857 2
732 0858 2 DIGIT = (.CHAR^4) + '0';
733 0859 2
734 0860 2 IF (.DIGIT GTR '9') THEN DIGIT = .DIGIT - 10 - '0' + 'A';
735 0861 2
736 0862 2 CHAR_REP [1] = .DIGIT;
737 0863 2 DIGIT = (.CHAR AND 15) + '0';
738 0864 2
739 0865 2 IF (.DIGIT GTR '9') THEN DIGIT = .DIGIT - 10 - '0' + 'A';
740 0866 2
741 0867 2 CHAR_REP [2] = .DIGIT;
742 0868 2
743 0869 2 END;
744 0870 2
745 0871 2
746 0872 2
```

747	0873	4
748	0874	4
749	0875	3
750	0876	3
751	0877	3
752	0878	3
753	0879	3
754	0880	3
755	0881	2
756	0882	2
757	0883	2
758	0884	2
759	0885	2
760	0886	1

```
CHAR_REP [3] = '>';
CHAR_DESC [DSC$W_LENGTH] = 4;
END;

STATUS = STR$CONCAT (INTER_DESC, INTER_DESC, CHAR_DESC);
IF ( NOT .STATUS) THEN RETURN (.STATUS);
END;

STATUS = STR$COPY DX (.OUTPUT_DESC, INTER_DESC);
DISCARD DESCRIPTOR (INTER_DESC);
RETURN T.STATUS);
END;

! End of routine PRINTABLE TEXT
```

PC	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419
----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

Sym

CTL
CTL
EXE
EXE
EXE
EXE
EXE
EXE
EXE
FPS
FPS
FPS
FPS
FPS
FP
MMG
MMG
PRS
SYS
VAX
VAX

EDT\$WRIEDTMSG
V04-000

EDT\$WRIEDTMSG - write VMSMSG.MSG
PRINTABLE_TEXT - Return a binary string in ASCII

N 4
16-Sep-1984 02:18:31
14-Sep-1984 12:25:55

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]WRIEDTMSG.B32;1

Page 27
(8)

		39		50	D1	00087		CMPL	DIGIT, #57		0868
				03	15	0008A		BLEQ	5\$		
		50		07	C0	0008C		ADDL2	#7, DIGIT		
	02	AE		50	90	0008F	5\$:	MOVB	DIGIT, CHAR REP+2		0870
	03	AE		3E	90	00093		MOVB	#62, CHAR REP+3		0873
	04	AE		04	B0	00097		MOVW	#4, CHAR DESC		0874
			04	AE	9F	0009B	6\$:	PUSHAB	CHAR DESC		0877
			10	AE	9F	0009E		PUSHAB	INTER_DESC		
			14	AE	9F	000A1		PUSHAB	INTER_DESC		
		00000000G	00	03	FB	000A4		CALLS	#3, STR\$CONCAT		
			53	50	D0	000AB		MOVL	R0, STATUS		
			2D	53	E9	000AE		BLBC	STATUS, 8\$		0879
FF69			01	08	AC	F1	000B1	7\$:	ACBL	INPUT_LEN, #1, CHAR_NO, 1\$	0820
	52			0C	AE	9F	000B8		PUSHAB	INTER_DESC	0883
				04	AC	DD	000BB		PUSHL	OUTPUT_DESC	
		00000000G	00	02	FB	000BE		CALLS	#2, STR\$COPY_DX		
			53	50	D0	000C5		MOVL	R0, STATUS		
				0C	AE	9F	000C8		PUSHAB	INTER_DESC	0884
		00000000G	00	01	FB	000CB		CALLS	#1, STR\$FREE1 DX		
			09	50	E8	000D2		BLBS	FREE_STATUS, 8\$		
				50	DD	000D5		PUSHL	FREE_STATUS		
		00000000G	00	01	FB	000D7		CALLS	#1, CIB\$STOP		
			50	53	D0	000DE	8\$:	MOVL	STATUS, R0		0885
					04	000E1		RET			0886

; Routine Size: 226 bytes, Routine Base: _EDT\$CODE + 0693

; 761 0887 1 !<BLF/PAGE>

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	1909	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	106	1	581	00:02.4

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACEBACK/LIS=LIS\$:WRIEDTMSG/OBJ=OBJ\$:WRIEDTMSG MSRC\$:WRIEDTMSG.B32/UPDATE=(ENH\$:WRIEDTMSG)

Size: 1601 code + 308 data bytes

Run Time: 01:14.7

Elapsed Time: 02:10.7

Lines/CPU Min: 714

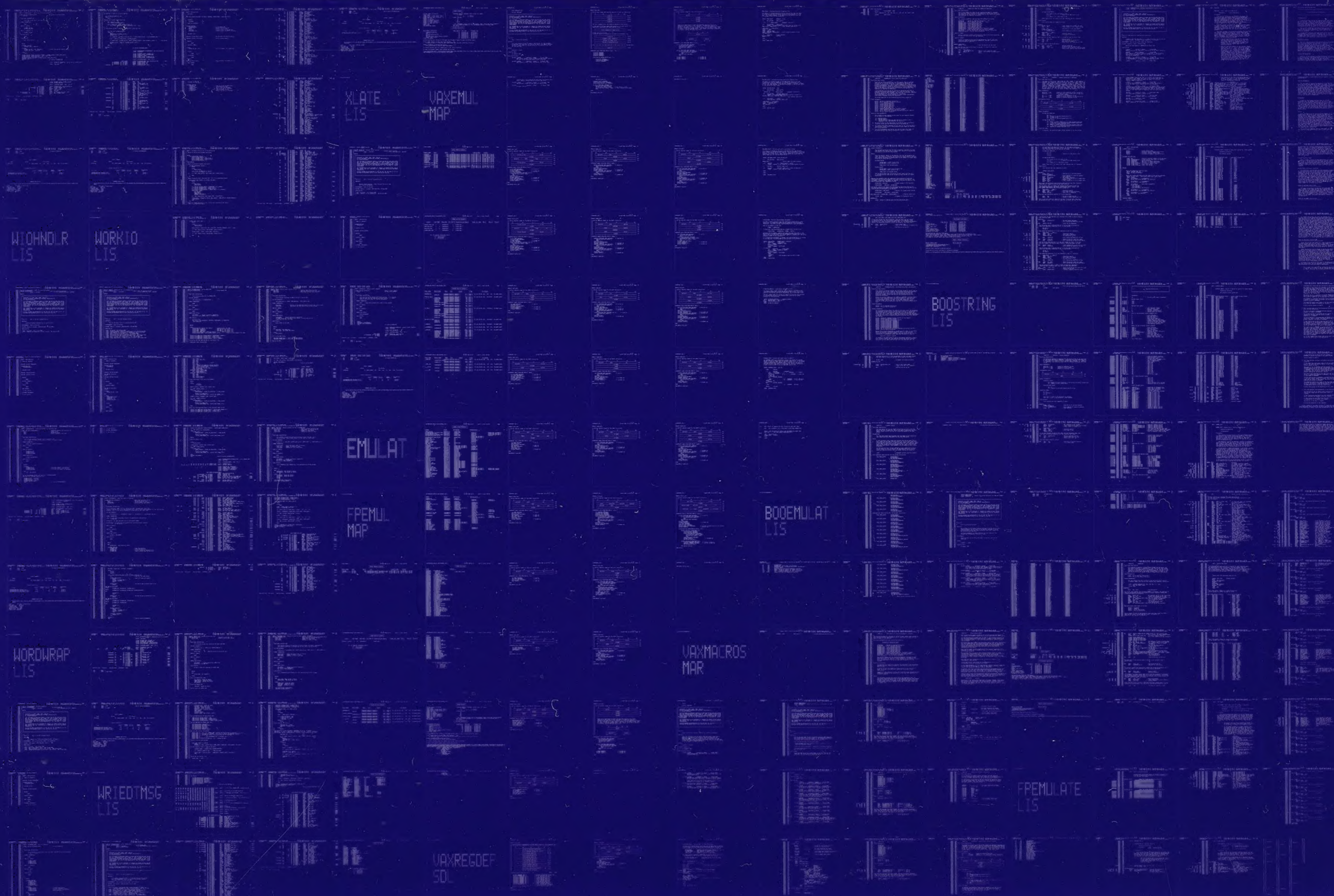
Lexemes/CPU-Min: 11820

Memory Used: 334 pages

Compilation Complete

0142 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



WIOHNDLR
LIS

WORKIO
LIS

XLATE
LIS

VAXEMUL
MAP

BOOSTRING
LIS

EMULAT

FPEMUL
MAP

BOOEMULAT
LIS

WORDWRAP
LIS

VAXMACROS
MAR

WRTEOTMSG
LIS

FPEMULATE
LIS

VAXREGDEF
SOL